Math Article

A small contribution to mathematics: The companion matrix

Determination of the characteristic polynomial by bilateral elimination

Sergej Oks

Math Article

A small contribution to mathematics: The companion matrix

Determination of the characteristic polynomial by bilateral elimination

Sergej Oks

12th January 2026

Contents

1 Intention **1**

2 Bilateral elimination **3**
2.1 Construction of the transformation matrix 3
2.2 Solving procedure 6
2.3 A short example calculation 9

3 Conclusion **11**

4 Appendix **13**
4.1 Matlab code: Companion matrix (original) 13
4.2 Matlab code: Companion matrix (optimized) 14
4.3 Matlab code: Companion matrix (Danilewski) 15

References and note on creation **16**

1 Intention

This work aims to document the possibility of determining the characteristic polynomial $p_A(\lambda)$ by transforming a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ into a so-called companion matrix $\tilde{\boldsymbol{A}}$. The calculation of the corresponding eigenvalues λ_i with $i = 1$ to n is explicitly not the focus here. The companion matrix is structured such that the subdiagonal contains ones and the last column contains the (negative) coefficients of the characteristic polynomial.

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \tag{1.1}$$

$$\Rightarrow \tilde{\boldsymbol{A}} = \text{transform}(\boldsymbol{A}) = \begin{bmatrix} 0 & 0 & \dots & & \dots & 0 & \tilde{A}_{1n} \\ 1 & 0 & \dots & & \dots & 0 & \tilde{A}_{2n} \\ 0 & 1 & \ddots & & & \vdots & \vdots \\ \vdots & \ddots & \ddots & & & & \\ & & & & & & \\ \vdots & & & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & & \ddots & 1 & 0 & \tilde{A}_{(n-1)n} \\ 0 & 0 & \dots & \dots & 0 & 1 & \tilde{A}_{nn} \end{bmatrix} \tag{1.2}$$

These results are obtained automatically during the calculation. It should be noted that, alternatively, a transposed companion matrix can be determined, in which the coefficients would accordingly appear in the last row. Different literature presents different methods of representation.

The characteristic polynomial is given by $\det(\lambda\boldsymbol{I} - \boldsymbol{A}) = \det(\lambda\boldsymbol{I} - \tilde{\boldsymbol{A}})$ which results in:

$$p_A(\lambda) = p_{\tilde{A}}(\lambda) = \lambda^n - \tilde{A}_{nn}\lambda^{n-1} - \tilde{A}_{(n-1)n}\lambda^{n-2} - \dots - \tilde{A}_{2n}\lambda^1 - \tilde{A}_{1n}\lambda^0 \tag{1.3}$$

And by setting the polynomial to zero $p_A(\lambda) = p_{\tilde{A}}(\lambda) = 0$, one obtains the characteristic equation for determining the eigenvalues:

$$\lambda^n = \tilde{A}_{nn}\lambda^{n-1} + \tilde{A}_{(n-1)n}\lambda^{n-2} + \dots + \tilde{A}_{2n}\lambda^1 + \tilde{A}_{1n}\lambda^0 \tag{1.4}$$

To determine the companion matrix, it is necessary to preserve the fundamental properties of a matrix during its transformation. Specifically, this involves preserving the trace, and thus the sum of all eigenvalues, as well as the determinant, and thus the product of all eigenvalues. In particular, the following must hold:

$$\operatorname{trace}(\boldsymbol{A}) = \operatorname{trace}(\tilde{\boldsymbol{A}}) = \tilde{A}_{nn} \tag{1.5}$$

And taking into account the alternating sign depending on the dimension n:

$$\det(\boldsymbol{A}) = \det(\tilde{\boldsymbol{A}}) = (-1)^{n+1}\tilde{A}_{1n} \tag{1.6}$$

The invariance of these two properties is given by the so-called similarity transformation. As already mentioned, an explicit calculation of the eigenvalues is omitted. This is possible with the so-called Danilewski method [1], which will be discussed in a slightly modified form in the following chapter.

2 Bilateral elimination

Bilateral elimination is achieved by the stepwise application of similarity transformations. While unilateral elimination (such as in the Gaussian elimination) involves a one-sided multiplication by an elementary matrix to perform either row or column operations, bilateral elimination (such as in the Danilewski method) balances each row operation with a corresponding column operation. Specifically, a row operation results from a left-hand multiplication

$$\tilde{\boldsymbol{A}}_{Rows} = \boldsymbol{L} \cdot \boldsymbol{A} \tag{2.1}$$

and a column operation resulting from a right-hand multiplication.

$$\tilde{\boldsymbol{A}}_{Columns} = \boldsymbol{A} \cdot \boldsymbol{R} \tag{2.2}$$

The two-sided multiplication in a similarity transformation manipulates both rows and columns. This makes it somewhat more difficult to find a suitable transformation matrix $\boldsymbol{P}$ with which zeros can be generated.

$$\tilde{\boldsymbol{A}} = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.3}$$

Furthermore, inverting the transformation matrix on one side also results in the inverse of the corresponding operation. An addition (or sum) thus necessitates a subtraction (or difference), which preserves the trace, and a multiplication is combined with a division, guaranteeing determinant preservation. As already clarified in chapter 1, these properties are essential for preserving the eigenvalues and thus the characteristic polynomial of the matrix $\boldsymbol{A}$. The invariance can be readily seen by rearranging the equation 2.3.

$$\Leftrightarrow \boldsymbol{P} \cdot \tilde{\boldsymbol{A}} = \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.4}$$

It becomes quite clear that the matrices $\boldsymbol{A}$ and $\tilde{\boldsymbol{A}}$ must be essentially the same and differ only in appearance due to the different application of the transformation matrix $\boldsymbol{P}$ (left-hand side as a row operation and right-hand side as a column operation).

2.1 Construction of the transformation matrix

The transformation matrix $\boldsymbol{P}$ for a position x in $i = 1$ to n can be formed from an elementary matrix for multiplication

$$\boldsymbol{P}_{mult} = \begin{bmatrix} P_{11} & 0 & \dots & & \dots & & \dots & 0 & 0 \\ 0 & P_{22} & \ddots & & & & & 0 & 0 \\ \vdots & \ddots & \ddots & & & & & & \vdots \\ & & & P_{(x-1)(x-1)} & 0 & & & & \\ \vdots & & & 0 & P_{xx} & 0 & & & \vdots \\ & & & & 0 & P_{(x+1)(x+1)} & & & \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & 0 & & & & & \ddots & P_{(n-1)(n-1)} & 0 \\ 0 & 0 & \dots & & \dots & & \dots & 0 & P_{nn} \end{bmatrix} \tag{2.5}$$

or division

$$\boldsymbol{P}_{div} = \boldsymbol{P}_{mult}^{-1} = \begin{bmatrix} \frac{1}{P_{11}} & 0 & \dots & & \dots & & \dots & 0 & 0 \\ 0 & \frac{1}{P_{22}} & \ddots & & & & & 0 & 0 \\ \vdots & \ddots & \ddots & & & & & & \vdots \\ & & & \frac{1}{P_{(x-1)(x-1)}} & 0 & & & & \\ \vdots & & & 0 & \frac{1}{P_{xx}} & 0 & & & \vdots \\ & & & & 0 & \frac{1}{P_{(x+1)(x+1)}} & & & \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & 0 & & & & & \ddots & \frac{1}{P_{(n-1)(n-1)}} & 0 \\ 0 & 0 & \dots & & \dots & & \dots & 0 & \frac{1}{P_{nn}} \end{bmatrix} \tag{2.6}$$

in combination with an elementary matrix for simple addition with the all-ones vector at the position x

$$\boldsymbol{P}_{add} = \begin{bmatrix} 1 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \ddots & 0 & 1 & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ & & & 1 & 1 & 0 & & & \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ & & & 0 & 1 & 1 & & & \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & 1 & 0 & \ddots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{2.7}$$

or the corresponding elementary matrix for the difference.

$$\boldsymbol{P}_{diff} = \boldsymbol{P}_{add}^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 & -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \ddots & 0 & -1 & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ & & & 1 & -1 & 0 & & & \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ & & & 0 & -1 & 1 & & & \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & -1 & 0 & \ddots & 1 & 0 \\ 0 & 0 & \dots & 0 & -1 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{2.8}$$

The matrices listed here, each for a separate conversion step, can in principle be combined in different ways (including transposed versions), but it is important to ensure that the order of the operations generates zeros at certain positions. Therefore, multiplication/division must be performed first, followed by addition/subtraction (as in Gaussian elimination). In this paper, the following combination will be selected

$$\boldsymbol{P} = \boldsymbol{P}_{mult} \cdot \boldsymbol{P}_{add} \tag{2.9}$$

$$\Rightarrow \boldsymbol{P} = \begin{bmatrix} P_{11} & 0 & \dots & 0 & P_{11} & 0 & \dots & 0 & 0 \\ 0 & P_{22} & \ddots & 0 & P_{22} & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ \vdots & & & P_{(x-1)(x-1)} & P_{(x-1)(x-1)} & 0 & & & \vdots \\ \vdots & & & 0 & P_{xx} & 0 & & & \vdots \\ \vdots & & & 0 & P_{(x+1)(x+1)} & P_{(x+1)(x+1)} & & & \vdots \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & P_{(n-1)(n-1)} & 0 & \ddots & P_{(n-1)(n-1)} & 0 \\ 0 & 0 & \dots & 0 & P_{nn} & 0 & \dots & 0 & P_{nn} \end{bmatrix} \tag{2.10}$$

and thus also the corresponding inverse.

$$\boldsymbol{P}^{-1} = \begin{bmatrix} \frac{1}{P_{11}} & 0 & \dots & 0 & -\frac{1}{P_{xx}} & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{P_{22}} & \ddots & 0 & -\frac{1}{P_{xx}} & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ \vdots & & & \frac{1}{P_{(x-1)(x-1)}} & -\frac{1}{P_{xx}} & 0 & & & \vdots \\ \vdots & & & 0 & \frac{1}{P_{xx}} & 0 & & & \vdots \\ \vdots & & & 0 & -\frac{1}{P_{xx}} & \frac{1}{P_{(x+1)(x+1)}} & & & \vdots \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & -\frac{1}{P_{xx}} & 0 & \ddots & \frac{1}{P_{(n-1)(n-1)}} & 0 \\ 0 & 0 & \dots & 0 & -\frac{1}{P_{xx}} & 0 & \dots & 0 & \frac{1}{P_{nn}} \end{bmatrix} \tag{2.11}$$

The resulting transformation matrix includes a diagonal and a vector at position x with the factors $P_{ii} \neq 0$ for $i = 1$ to n. All other entries are zero. The simplicity of the inverse allows it to be constructed directly without having to determine it algorithmically, thus reducing computational effort.

2.2 Solving procedure

The transformation matrix $\boldsymbol{P}$ explained in the previous chapter can now be used for a given matrix

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ A_{31} & A_{32} & \dots & A_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \tag{2.12}$$

to determine a solution step by step. From the components A_{ik} in the first step, $k = 1$, the entries of the vector $\boldsymbol{A}_{i1}$ are transferred to the corresponding positions P_{ii} in the transformation matrix. In addition to the values on the main diagonal, $\boldsymbol{P}$ contains the complete vector $\boldsymbol{A}_{i1}$ at the position $x = k + 1 = 2$.

$$\boldsymbol{P} = \begin{bmatrix} A_{11} & A_{11} & 0 & 0 & \dots & 0 \\ 0 & A_{21} & 0 & 0 & \dots & 0 \\ 0 & A_{31} & A_{31} & 0 & \dots & 0 \\ 0 & A_{41} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & A_{(n-1)1} & 0 \\ 0 & A_{n1} & 0 & \dots & 0 & A_{n1} \end{bmatrix} \tag{2.13}$$

The corresponding inverse is:

$$\boldsymbol{P}^{-1} = \begin{bmatrix} 1/A_{11} & -1/A_{21} & 0 & 0 & \dots & 0 \\ 0 & 1/A_{21} & 0 & 0 & \dots & 0 \\ 0 & -1/A_{21} & 1/A_{31} & 0 & \dots & 0 \\ 0 & -1/A_{21} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1/A_{(n-1)1} & 0 \\ 0 & -1/A_{21} & 0 & \dots & 0 & 1/A_{n1} \end{bmatrix} \tag{2.14}$$

Applying this transformation means that in the first step, the first column $k = 1$ of the matrix $\boldsymbol{A}$ is processed, but the only non-zero entry in the second row of this column remains. Ultimately, the goal is to create a subdiagonal, which implies the starting value $x = 2$, and from which it follows that the solution algorithm can execute a maximum of $(n-1)$ steps. If it is not immediately obvious that the starting value cannot be $x = 1$, one must remember that this is not about the main diagonal. The main diagonal would contain the eigenvalues, which, due to their non-linear properties, cannot be determined in this relatively simple way. One obtains

$$\boldsymbol{A}' = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} = \begin{bmatrix} 0 & A'_{12} & \dots & A'_{1n} \\ A'_{21} & A'_{22} & \dots & A'_{2n} \\ 0 & A'_{32} & \dots & A'_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & A'_{n2} & \dots & A'_{nn} \end{bmatrix} \tag{2.15}$$

and can continue with the next step $k = 2$ and therefore $x = 3$. The corresponding transformation matrix is given by

$$\boldsymbol{P}' = \begin{bmatrix} A'_{12} & 0 & A'_{12} & 0 & 0 & \dots & 0 \\ 0 & A'_{22} & A'_{22} & 0 & 0 & \dots & 0 \\ 0 & 0 & A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & A'_{42} & A'_{42} & 0 & \dots & 0 \\ 0 & 0 & A'_{52} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & A'_{(n-1)2} & 0 \\ 0 & 0 & A'_{n2} & 0 & \dots & 0 & A'_{n2} \end{bmatrix} \tag{2.16}$$

with the corresponding inverse

$$\boldsymbol{P}'^{-1} = \begin{bmatrix} 1/A'_{12} & 0 & -1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 1/A'_{22} & -1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & 1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & -1/A'_{32} & 1/A'_{42} & 0 & \dots & 0 \\ 0 & 0 & -1/A'_{32} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 1/A'_{(n-1)2} & 0 \\ 0 & 0 & -1/A'_{32} & 0 & \dots & 0 & 1/A'_{n2} \end{bmatrix} \tag{2.17}$$

so that with the result

$$\boldsymbol{A}'' = \boldsymbol{P}'^{-1} \cdot \boldsymbol{A}' \cdot \boldsymbol{P}' = \begin{bmatrix} 0 & 0 & A''_{13} & \dots & A''_{1n} \\ A''_{21} & 0 & A''_{23} & \dots & A''_{2n} \\ 0 & A''_{32} & A''_{33} & \dots & A''_{3n} \\ 0 & 0 & A''_{43} & \dots & A''_{4n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & A''_{n3} & \dots & A''_{nn} \end{bmatrix} \tag{2.18}$$

further steps until the $(n-1)$ step

$$\boldsymbol{A}^{(n-1)} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & A_{1n}^{(n-1)} \\ A_{21}^{(n-1)} & 0 & 0 & \dots & 0 & A_{2n}^{(n-1)} \\ 0 & A_{32}^{(n-1)} & 0 & \dots & 0 & A_{3n}^{(n-1)} \\ 0 & 0 & A_{43}^{(n-1)} & \ddots & 0 & A_{4n}^{(n-1)} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & A_{n(n-1)}^{(n-1)} & A_{nn}^{(n-1)} \end{bmatrix} \tag{2.19}$$

can be done. Finally, the entries in the subdiagonal must be divided so that only ones remain. Since every multiplication or division in a row also has the inverse effect on the corresponding column, two subdiagonal elements are always connected. This results in a successive application of the operations $P_{ii} = P_{(i-1)(i-1)} \cdot A_{i(i-1)}$ from $i = 2$ to n, so that with the starting value $P_{11} = 1$, the following transformation matrix is obtained.

$$\boldsymbol{P}^{(n-1)} =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & A_{21}^{(n-1)} & 0 & 0 & \dots & 0 \\ 0 & 0 & A_{21}^{(n-1)} \cdot A_{32}^{(n-1)} & 0 & \dots & 0 \\ 0 & 0 & 0 & A_{21}^{(n-1)} \cdot A_{32}^{(n-1)} \cdot A_{43}^{(n-1)} & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & A_{21}^{(n-1)} \cdot \dots \cdot A_{n(n-1)}^{(n-1)} \end{bmatrix} \tag{2.20}$$

The companion matrix is calculated as usual with

$$\boldsymbol{A}^{(n)} = (\boldsymbol{P}^{(n-1)})^{-1} \cdot \boldsymbol{A}^{(n-1)} \cdot \boldsymbol{P}^{(n-1)} \tag{2.21}$$

to get it in its final form

$$\Leftrightarrow \tilde{\boldsymbol{A}} = \boldsymbol{A}^{(n)} = \begin{bmatrix} 0 & 0 & \dots & & \dots & 0 & \tilde{A}_{1n} \\ 1 & 0 & \dots & & \dots & 0 & \tilde{A}_{2n} \\ 0 & 1 & \ddots & & & \vdots & \vdots \\ \vdots & \ddots & \ddots & & & & \\ & & & & & & \\ \vdots & & & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & & \ddots & 1 & 0 & \tilde{A}_{(n-1)n} \\ 0 & 0 & \dots & \dots & 0 & 1 & \tilde{A}_{nn} \end{bmatrix} \tag{2.22}$$

so that the characteristic polynomial, as described in chapter 1, can be read off here.

2.3 A short example calculation

For illustration, a simplified 3×3 example will be worked through in this section, in which the first step essentially is already obvious from the previous section. It suffices to read the first three rows and columns from the corresponding matrices and calculate the result. For a given matrix

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & A_{11} \\ A_{21} & A_{22} & A_{22} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \tag{2.23}$$

the first step results in:

$$\boldsymbol{A}' = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.24}$$

$$\Rightarrow \boldsymbol{A}' = \begin{bmatrix} 1/A_{11} & -1/A_{21} & 0 \\ 0 & 1/A_{21} & 0 \\ 0 & -1/A_{21} & 1/A_{31} \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} & A_{11} \\ A_{21} & A_{22} & A_{22} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{11} & 0 \\ 0 & A_{21} & 0 \\ 0 & A_{31} & A_{31} \end{bmatrix} \tag{2.25}$$

$$\Leftrightarrow \boldsymbol{A}' = \begin{bmatrix} 0 & A_{21}(\frac{A_{12}}{A_{11}} - \frac{A_{22}}{A_{21}}) + A_{31}(\frac{A_{13}}{A_{11}} - \frac{A_{23}}{A_{21}}) & A_{31}(\frac{A_{13}}{A_{11}} - \frac{A_{23}}{A_{21}}) \\ A_{11} & A_{11} + A_{22} + \frac{A_{23}A_{31}}{A_{21}} & \frac{A_{23}A_{31}}{A_{21}} \\ 0 & -A_{21}(\frac{A_{22}}{A_{21}} - \frac{A_{32}}{A_{31}}) - A_{31}(\frac{A_{23}}{A_{21}} - \frac{A_{33}}{A_{31}}) & A_{33} - \frac{A_{23}A_{31}}{A_{21}} \end{bmatrix} \tag{2.26}$$

Then the second step follows

$$\boldsymbol{A}'' = \boldsymbol{P}'^{-1} \cdot \boldsymbol{A}' \cdot \boldsymbol{P}' \tag{2.27}$$

and the additional final step:

$$\boldsymbol{A}''' = \boldsymbol{P}''^{-1} \cdot \boldsymbol{A}'' \cdot \boldsymbol{P}'' \tag{2.28}$$

The result can then be read from the last column, whereby only ones must remain in the subdiagonal.

$$\Rightarrow \boldsymbol{A}''' = \begin{bmatrix} 0 & 0 & A_{11}A_{22}A_{33} - A_{11}A_{23}A_{32} - A_{12}A_{21}A_{33} + A_{12}A_{23}A_{31} + A_{13}A_{21}A_{32} - A_{13}A_{22}A_{31} \\ 1 & 0 & A_{12}A_{21} - A_{11}A_{22} - A_{11}A_{33} + A_{13}A_{31} - A_{22}A_{33} + A_{23}A_{32} \\ 0 & 1 & A_{11} + A_{22} + A_{33} \end{bmatrix} \tag{2.29}$$

As can be seen, the trace of $\boldsymbol{A}$ was obtained in the bottom entry of the last column or in the last main diagonal element, while the determinant (with the corresponding sign) was determined in the top entry of the last column.

3 Conclusion

When determining the additional transformation matrix in equation 2.20 for the final step, it becomes clear that a numerical problem arises with sufficiently large matrices and entries. Therefore, while this method is useful for symbolic calculations in practice, it is not very practical for numerical computations. The conventional Danilewski method works somewhat more elegantly (see code example in the appendix 4.3), but its numerical instability is also well known. Therefore, although it may seem counterproductive, it makes sense to take a detour via an iterative calculation of the eigenvalues. A modern method would be, for example, the QR algorithm, which, although iterative, is significantly more numerically stable. Subsequently, the companion matrix or the characteristic polynomial can be derived from the eigenvalues, which represent the roots of the characteristic polynomial, if needed.

4 Appendix

4.1 Matlab code: Companion matrix (original)

```
function [C] = mcomp(A)
% Attention:  Suitable for symbolic calculation,
% but division by zero possible!

% Input:
    % A    % Matrix A

% Output:
    % C    % Companion matrix C

% Algorithm:
    C = A; n = size(C,1); I = eye(n);
    if isa(C,'sym')
            I = sym(I);
    end
    for j = 1 : n - 1
            x = j + 1;
            P_mult = diag(C(:, j));
            P_add = I; P_add(:, x) = 1;
            P = P_mult * P_add;
            C = P^-1 * C * P;
    end

% Closing step:
    P = I;
    for i = 2 : n
            P(i, i) = P(i - 1, i - 1) * C(i, i - 1);
    end
    C = P^-1 * C * P;
return
```

4.2 Matlab code: Companion matrix (optimized)

```
function [C] = mcomp(A)
% Attention:  Suitable for symbolic calculation,
% but division by zero possible!

% Input:
    % A   % Matrix A

% Output:
    % C   % Companion matrix C

% Algorithm:
    C = A; n = size(C,1);
    for j = 1 : n - 1
            x = j + 1;
            v = C(:, j);
            h = C(x, :)/v(x);
            s = v * 0;
            for i = 1 : n
                C(:, i) = (C(:, i)./v - h(i)) * v(i);
                C(x, i) = C(x, i) + h(i) * v(i);
                s = s + C(:, i);
            end
            C(:, x) = s;
    end

% Closing step:
    fact = 1;
    for i = 2 : n
            fact = fact * C(i, i - 1);
            C(i, n) = C(i, n)/fact;
            C(i, i - 1) = 1;
    end
    C(:, n) = C(:, n). * fact;
return
```

4.3 Matlab code: Companion matrix (Danilewski)

```
function [C] = mcomp(A)
% Attention:  Suitable for symbolic calculation,
% but division by zero possible!

% Input:
    % A    % Matrix A

% Output:
    % C    % Companion matrix C

% Algorithm:
    C = A; n = size(C,1); I = eye(n);
    if isa(C,'sym')
            I = sym(I);
    end
    for j = 1 : n - 1
            x = j + 1;
            P = I; P(:,x) = C(:,j);
            C = P^-1 * C * P;
    end
return
```

References

[1] Danilewski, A. M.: *On the numerical solution of the secular equation*

[2] Zurmühl, R. / Falk, S.: *Matrizen und ihre Anwendungen*

Note on creation

[3] Gemini AI (Google): *Support for better structuring and verification*